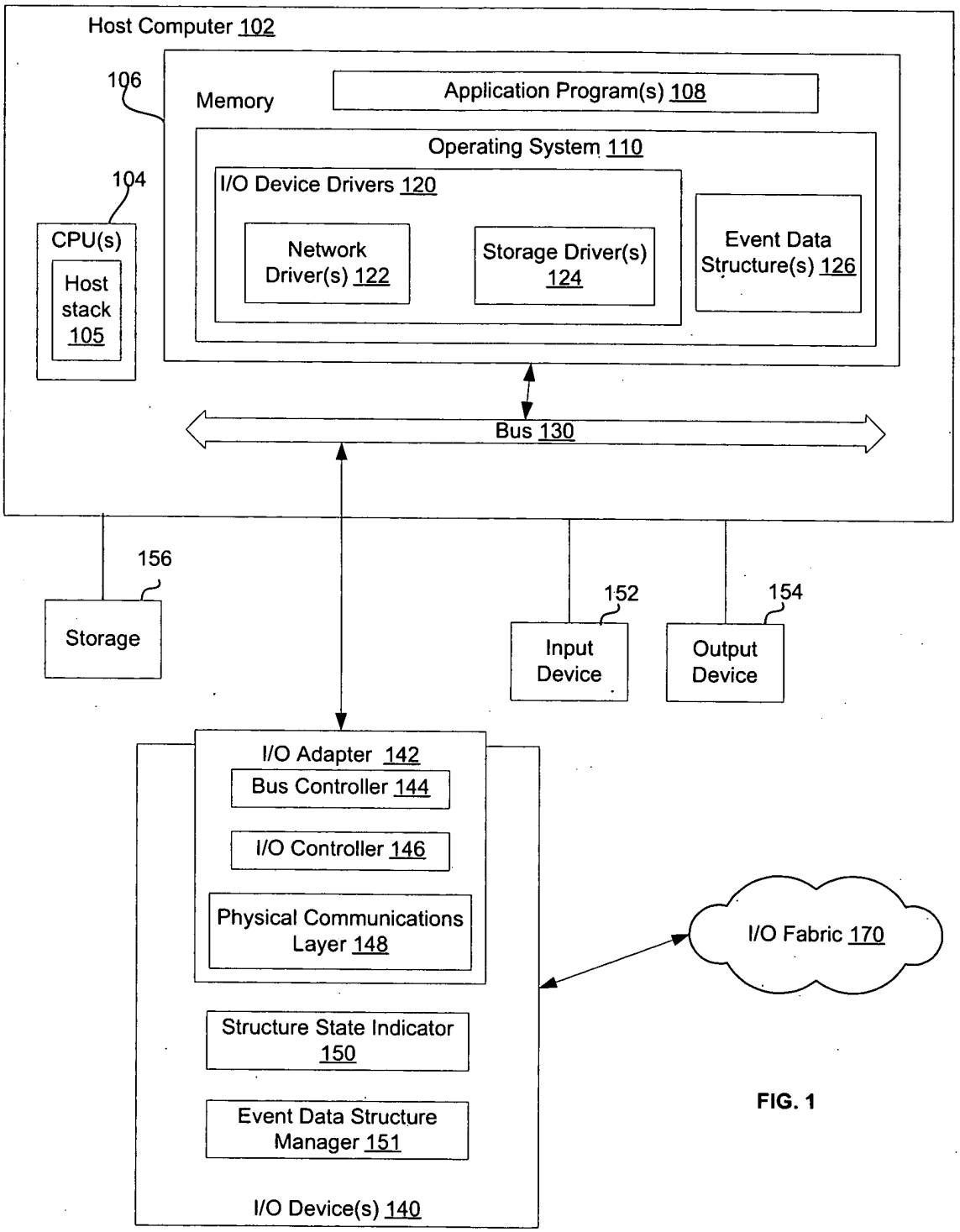
Host Computer 102
106
Memory
Application Program(s) 108
Operating System 110
104
I/O Device Drivers 120
CPU(s)
Host stack 105
Network Driver(s) 122
Storage Driver(s) 124
Event Data Structure(s) 126
Bus 130
156
Storage
152
Input Device
154
Output Device
I/O Adapter 142
Bus Controller 144
I/O Controller 146
Physical Communications Layer 148
I/O Fabric 170
Structure State Indicator 150
Event Data Structure Manager 151
I/O Device(s) 140

FIG. 1

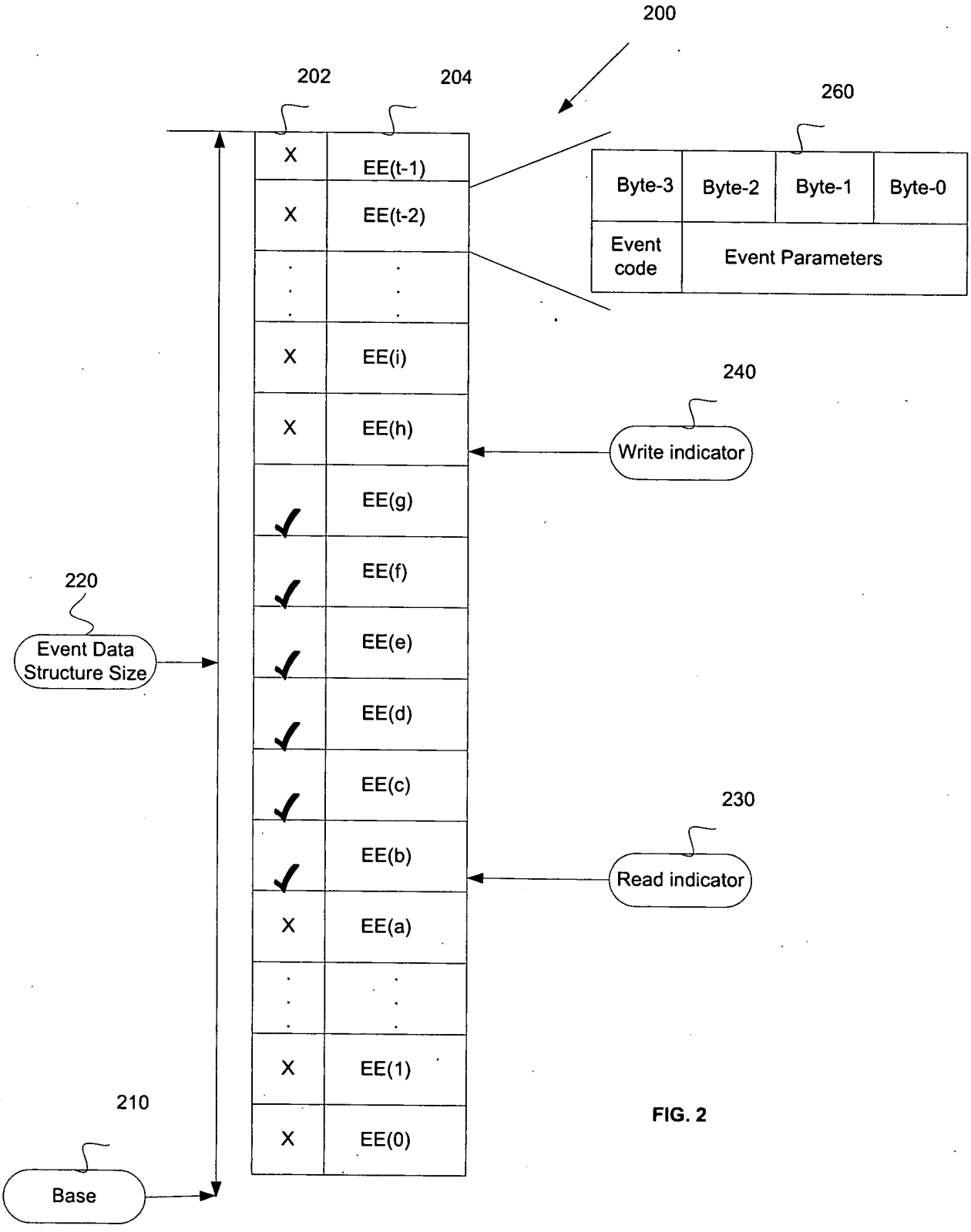
200
202
204
260
X
EE(t-1)
X
EE(t-2)
X
EE(i)
X
EE(h)
EE(g)
EE(f)
EE(e)
EE(d)
EE(c)
EE(b)
X
EE(a)
X
EE(1)
X
EE(0)
Byte-3
Byte-2
Byte-1
Byte-0
Event code
Event Parameters
240
Write indicator
220
Event Data Structure Size
230
Read indicator
210
Base

FIG. 2

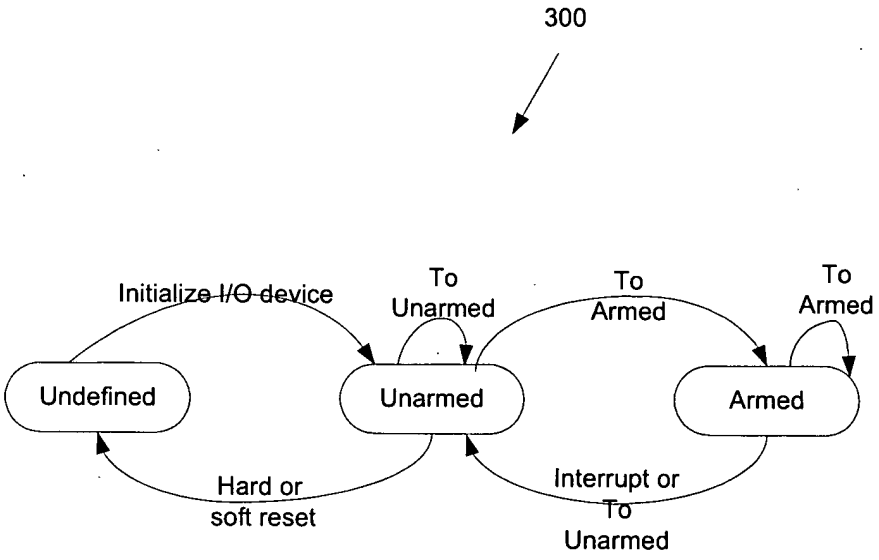
300
Initialize I/O device
To Unarmed
To Armed
To Armed
Undefined
Unarmed
Armed
Hard or soft reset
Interrupt or To Unarmed

FIG. 3

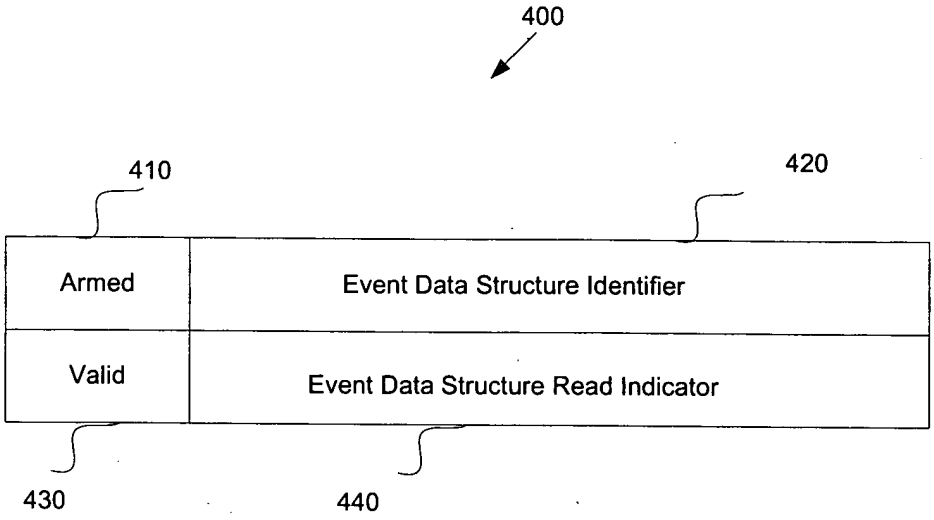
400
410
420
Armed
Event Data Structure Identifier
Valid
Event Data Structure Read Indicator
430
440

FIG. 4

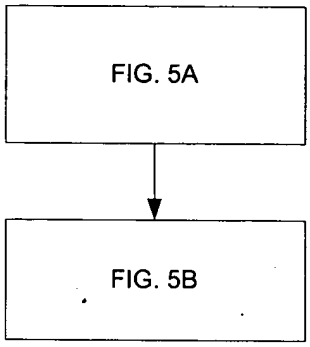

FIG. 5

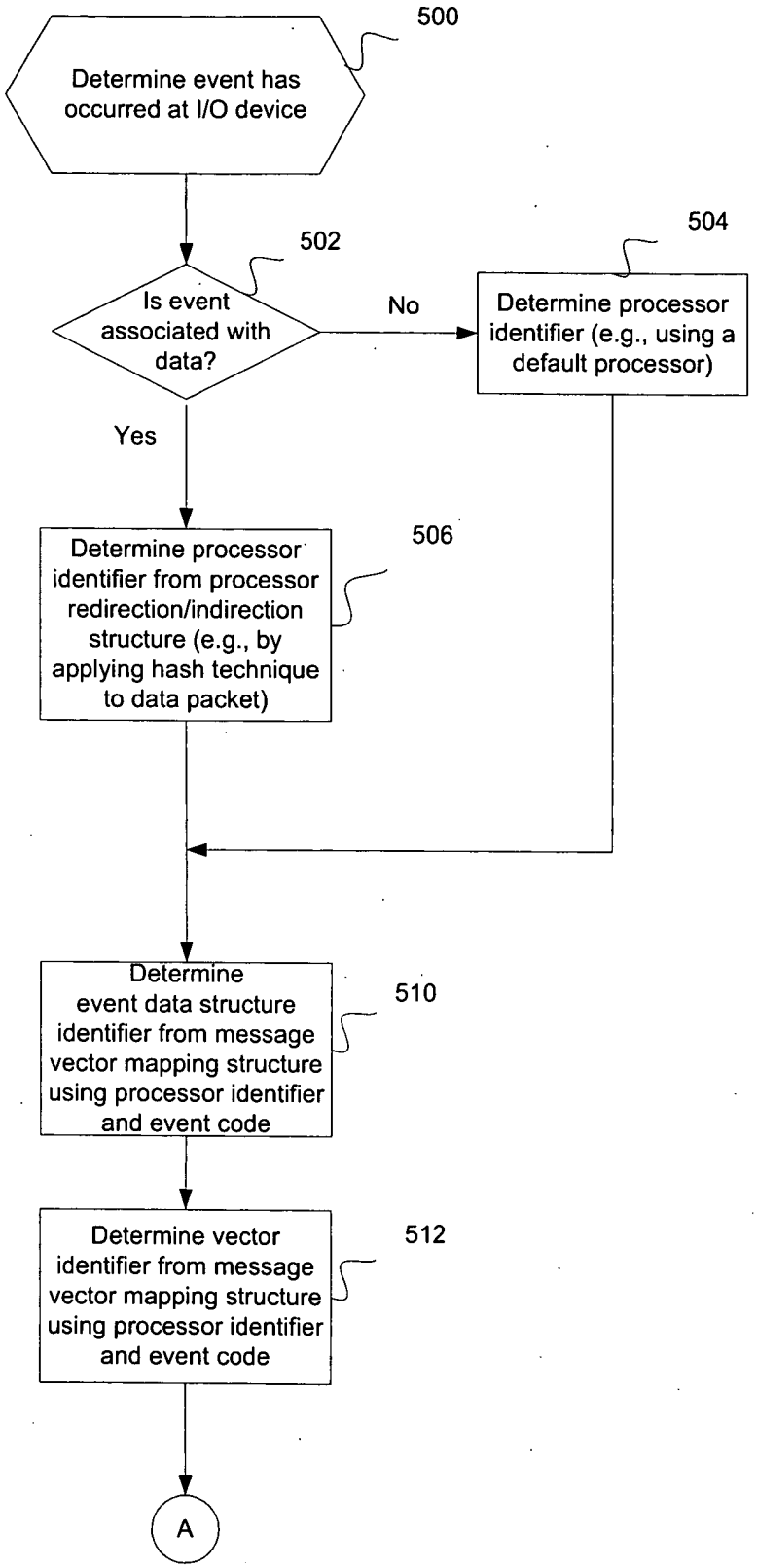
500
Determine event has occurred at I/O device
502
Is event associated with data?
No
504
Determine processor identifier (e.g., using a default processor)
Yes
506
Determine processor identifier from processor redirection/indirection structure (e.g., by applying hash technique to data packet)
510
Determine event data structure identifier from message vector mapping structure using processor identifier and event code
512
Determine vector identifier from message vector mapping structure using processor identifier and event code
A

FIG. 5A

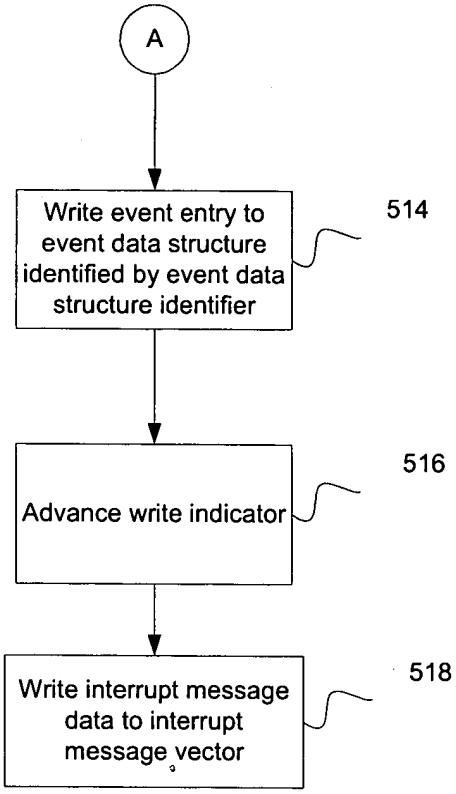
A
Write event entry to event data structure identified by event data structure identifier
514
Advance write indicator
516
Write interrupt message data to interrupt message vector
518

FIG. 5B

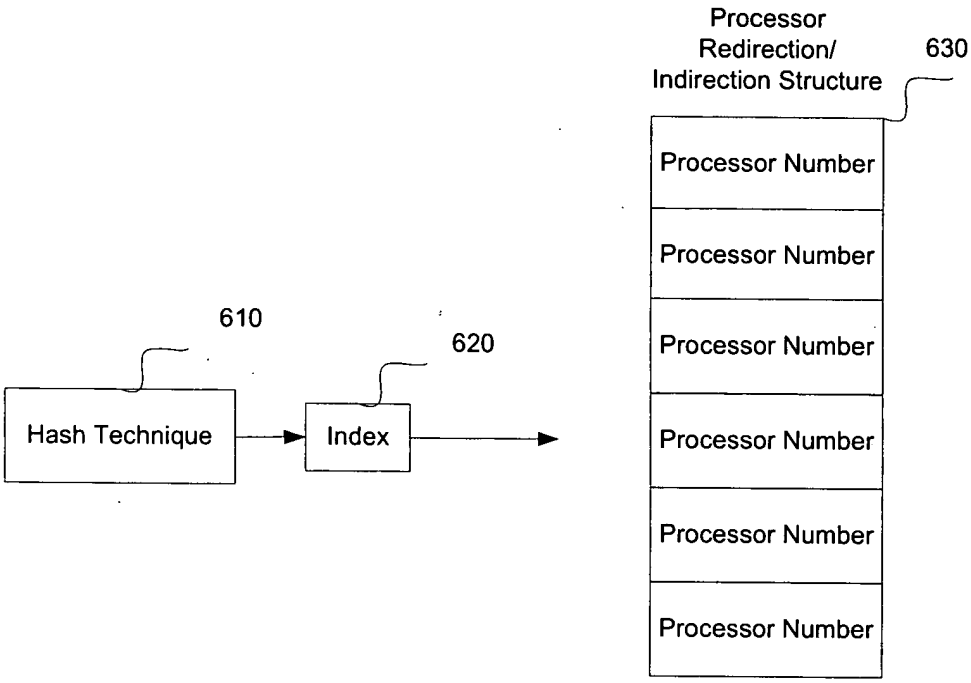
Processor
Redirection/
Indirection Structure
630
Processor Number
Processor Number
Processor Number
Processor Number
Processor Number
Processor Number
610
620
Hash Technique
Index

FIG. 6

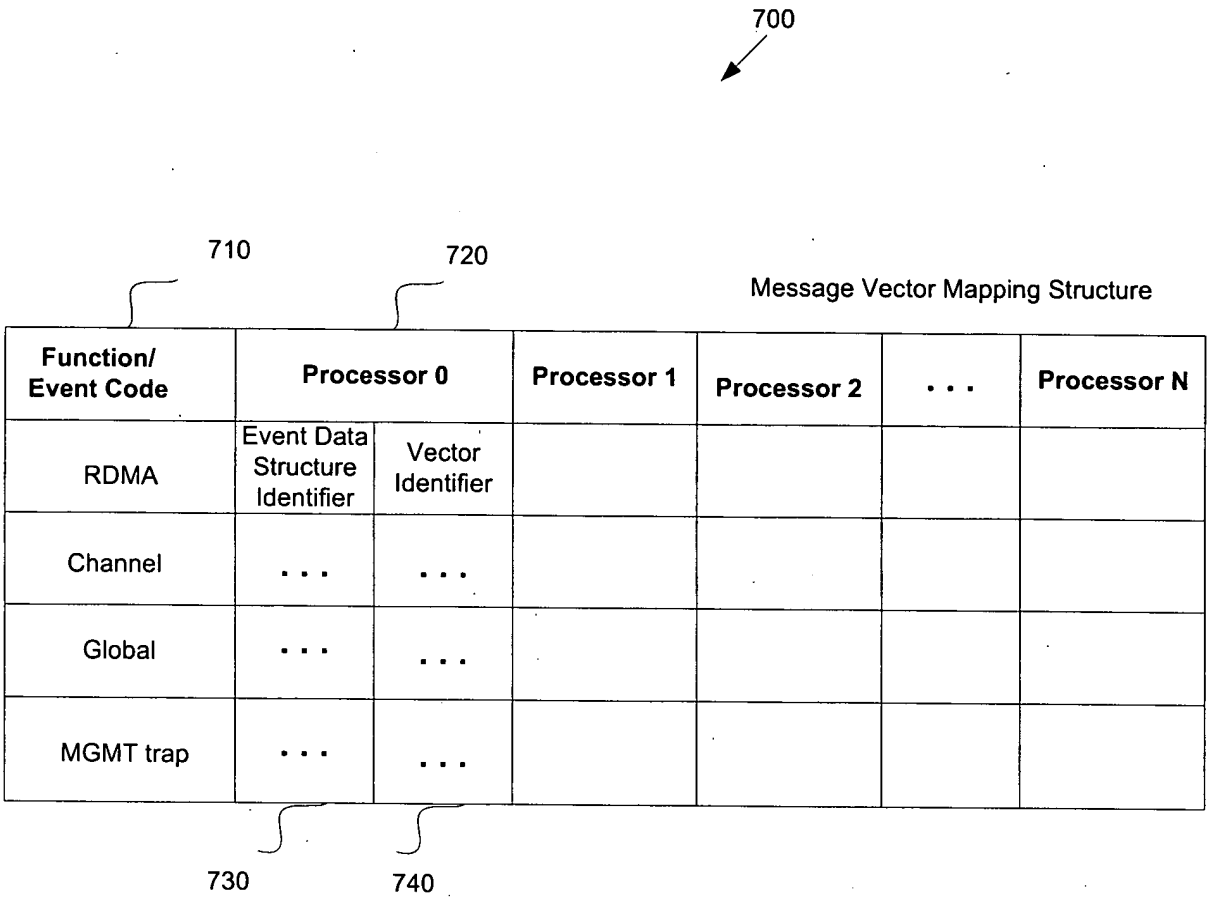
700
710
720
Message Vector Mapping Structure
Function/ Event Code
Processor 0
Processor 1
Processor 2
. . .
Processor N
RDMA
Event Data Structure Identifier
Vector Identifier
Channel
Global
MGMT trap
730
740

FIG. 7

860 870

Message Vector Mapping Structure

Function/ Event Code	Processor 0		Processor 1	Processor 2	. . .	Processor N
RDMA TX	Event Data Structure Identifier	Vector Identifier				
RDMA RX	Event Data Structure Identifier	Vector Identifier				
Channel TX	. . .	. . .				
Channel RX	. . .	. . .				
Global - General Purpose I/O (GPIO) Interrupts	. . .	. . .				
Global - Errors	. . .	. . .				
MGMT trap	. . .	. . .				

880 890

FIG. 8

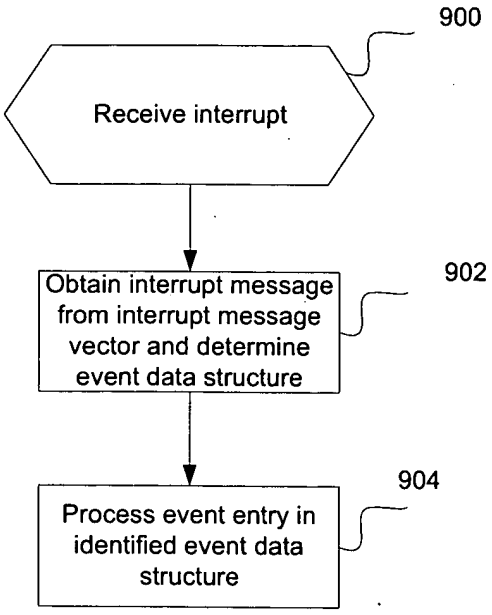
900
Receive interrupt
902
Obtain interrupt message from interrupt message vector and determine event data structure
904
Process event entry in identified event data structure

FIG. 9

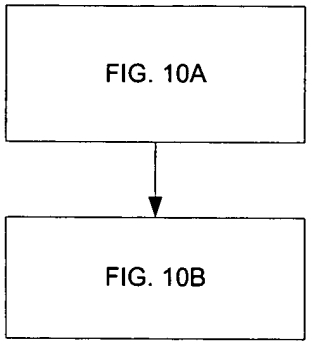

FIG. 10

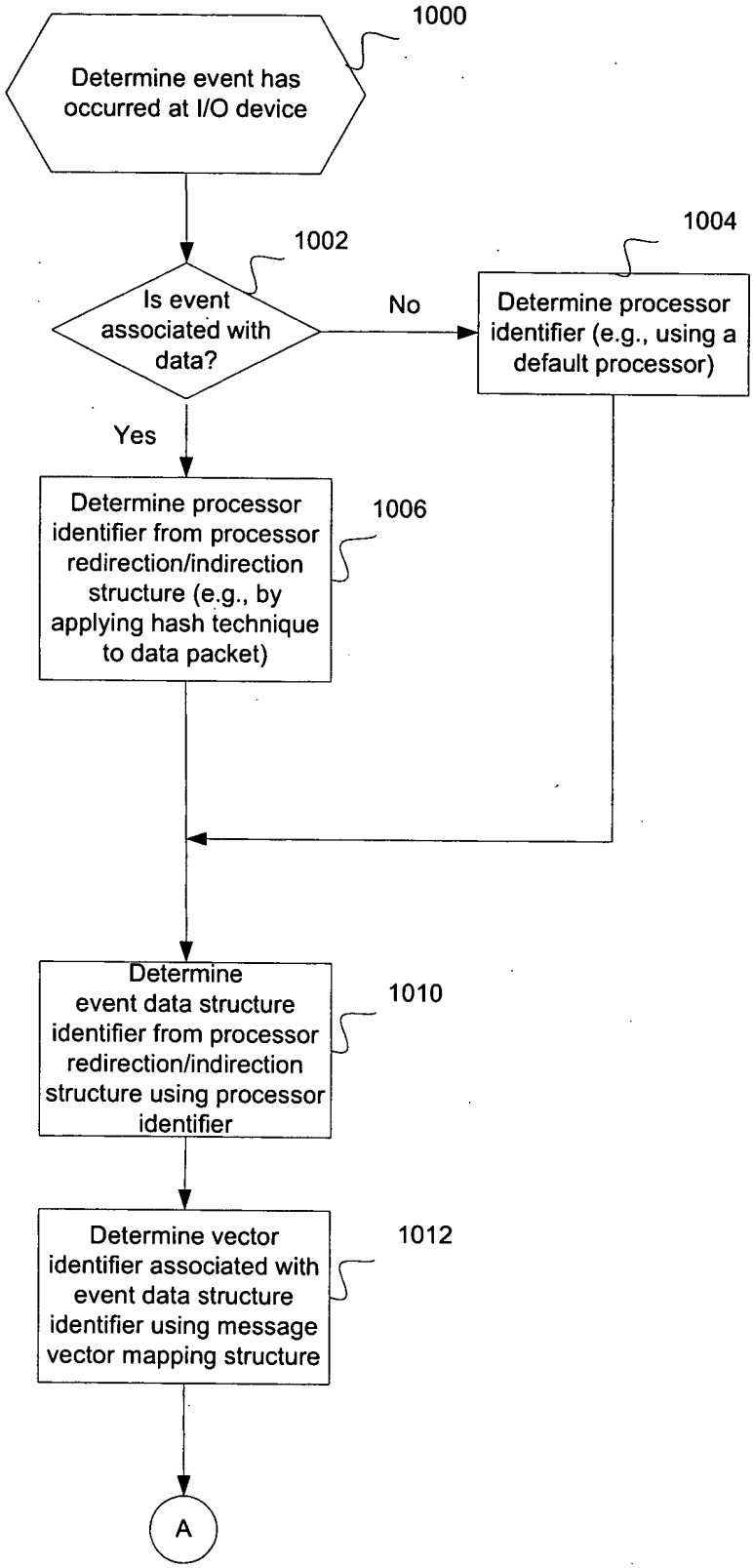
1000
Determine event has occurred at I/O device
1002
Is event associated with data?
No
1004
Determine processor identifier (e.g., using a default processor)
Yes
Determine processor identifier from processor redirection/indirection structure (e.g., by applying hash technique to data packet)
1006
Determine event data structure identifier from processor redirection/indirection structure using processor identifier
1010
Determine vector identifier associated with event data structure identifier using message vector mapping structure
1012
A

FIG. 10A

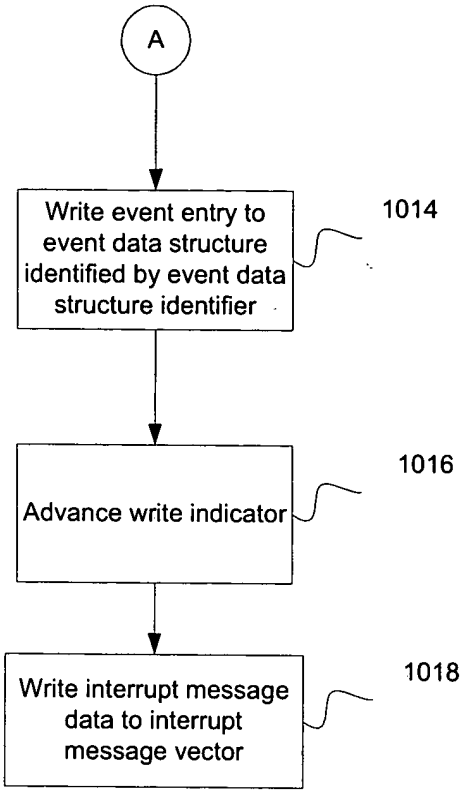
A
Write event entry to event data structure identified by event data structure identifier
1014
Advance write indicator
1016
Write interrupt message data to interrupt message vector
1018

FIG. 10B

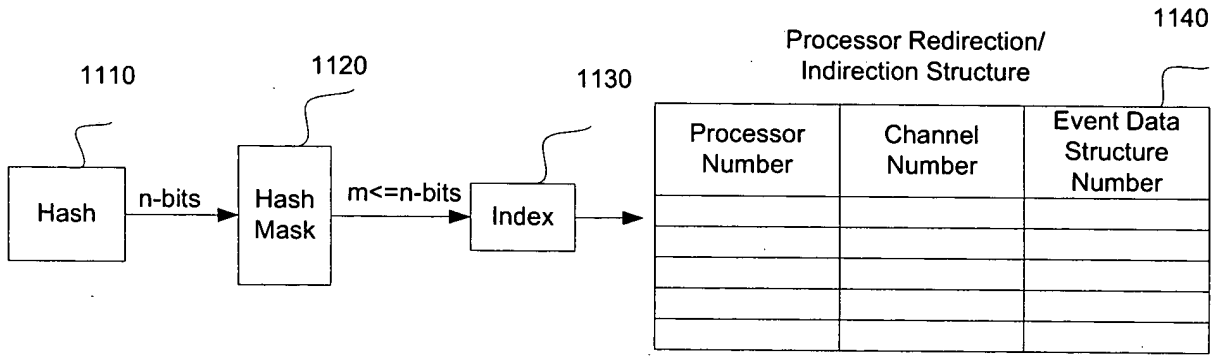
1110
1120
1130
1140
Processor Redirection/
Indirection Structure
Hash
n-bits
Hash
Mask
m<=n-bits
Index
Processor
Number
Channel
Number
Event Data
Structure
Number

FIG. 11

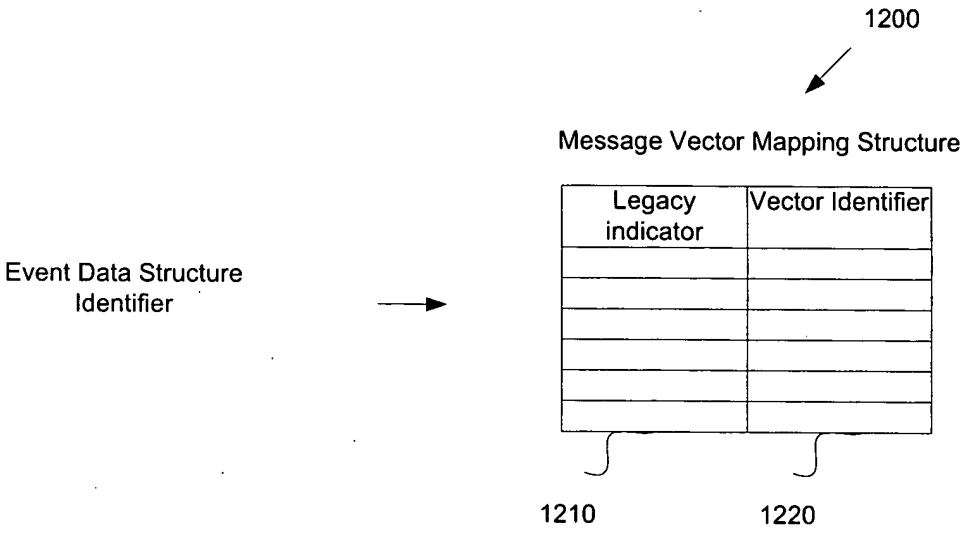
1200
Message Vector Mapping Structure
Legacy indicator
Vector Identifier
Event Data Structure Identifier
1210
1220

FIG. 12